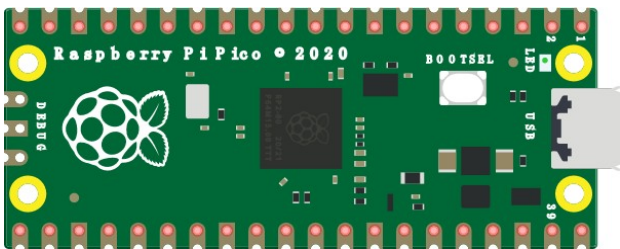

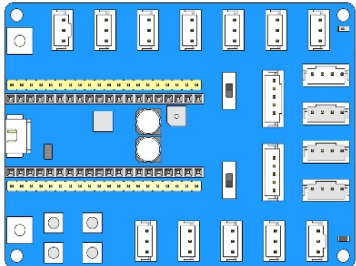
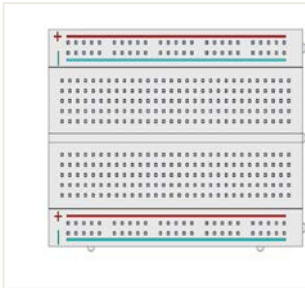





Lesson 2-Waterfall Light Led

In this lesson, we will use Raspberry Pi Pico and Pico Expansion Board to control 4 common Red LED working as waterfall light blinking.

PS: This kit does not provide red LEDs and 220 ohm resistors, if you refer to this lesson for experiments, you need to prepare 4 red LEDs and 4 220 ohm resistors by yourself

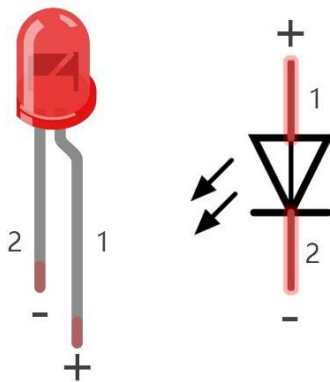
1.Component List

Raspberry Pi Pico x1		USB Cable x1
		
Pico Expansion board x1		Breadboard x1
		
LED x4	Resistor 220 Ω x4	Jumper
		

2.Component knowledge

LED

An LED is a type of diode. All diodes only work if current is flowing in the correct direction and have two Poles. An LED will only work (light up) if the longer pin (+) of LED is connected to the positive output from a power source and the shorter pin is connected to the negative (-). Negative output is also referred to as Ground (GND). This type of component is known as “Polar” (think One-Way Street). All common 2 lead diodes are the same in this respect. Diodes work only if the voltage of its positive electrode is higher than its negative electrode and there is a narrow range of operating voltage for most all common diodes of 1.9 and 3.4V. If you use much more than 3.3V the LED will be damaged and



LED	Voltage	Maximum current	Recommended current
Red	1.9-2.2V	20mA	10mA
Green	2.9-3.4V	10mA	5mA
Blue	2.9-3.4V	10mA	5mA
Volt ampere characteristics conform to diode			

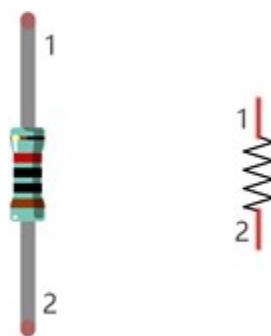
burn out.

Note: LEDs cannot be directly connected to a power supply, which usually ends in a damaged component. A resistor with a specified resistance value must be connected in series to the LED you plan to use.

Resistor

Resistors use Ohms (Ω) as the unit of measurement of their resistance (R). $1\text{M}\Omega=1000\text{k}\Omega$, $1\text{k}\Omega=1000\Omega$.

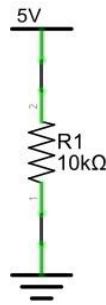
A resistor is a passive electrical component that limits or regulates the flow of current in an electronic circuit. On the left, we see a physical representation of a resistor, and the right is the symbol used to represent the presence of a resistor in a circuit diagram or schematic.



The bands of color on a resistor is a shorthand code used to identify its resistance value. For more details of resistor color codes, please refer to the appendix of this tutorial.

With a fixed voltage, there will be less current output with greater resistance added to the circuit. The relationship between Current, Voltage and Resistance can be expressed by this formula: $I=V/R$ known as Ohm's Law where I = Current, V = Voltage and R = Resistance. Knowing the values of any two of these allows you to solve the value of the third.

In the following diagram, the current through R1 is: $I=U/R=5V/10k\Omega=0.0005A=0.5mA$.

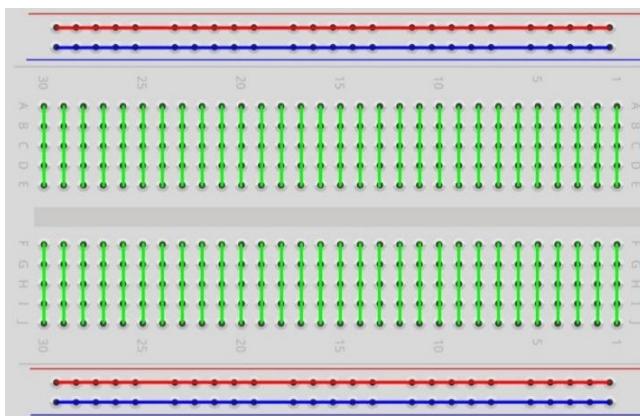


WARNING: Never connect the two poles of a power supply with anything of low resistance value (i.e. a metal object or bare wire) this is a Short and results in high current that may damage the power supply and electronic components.

Note: Unlike LEDs and Diodes, Resistors have no poles and are non-polar (it does not matter which direction you insert them into a circuit, it will work the same)

Breadboard

Here we have a small breadboard as an example of how the rows of holes (sockets) are electrically attached. The left picture shows the way to connect pins. The right picture shows the practical internal structure.



Power

The Pico Expansion Board should supply DC 7—12V.

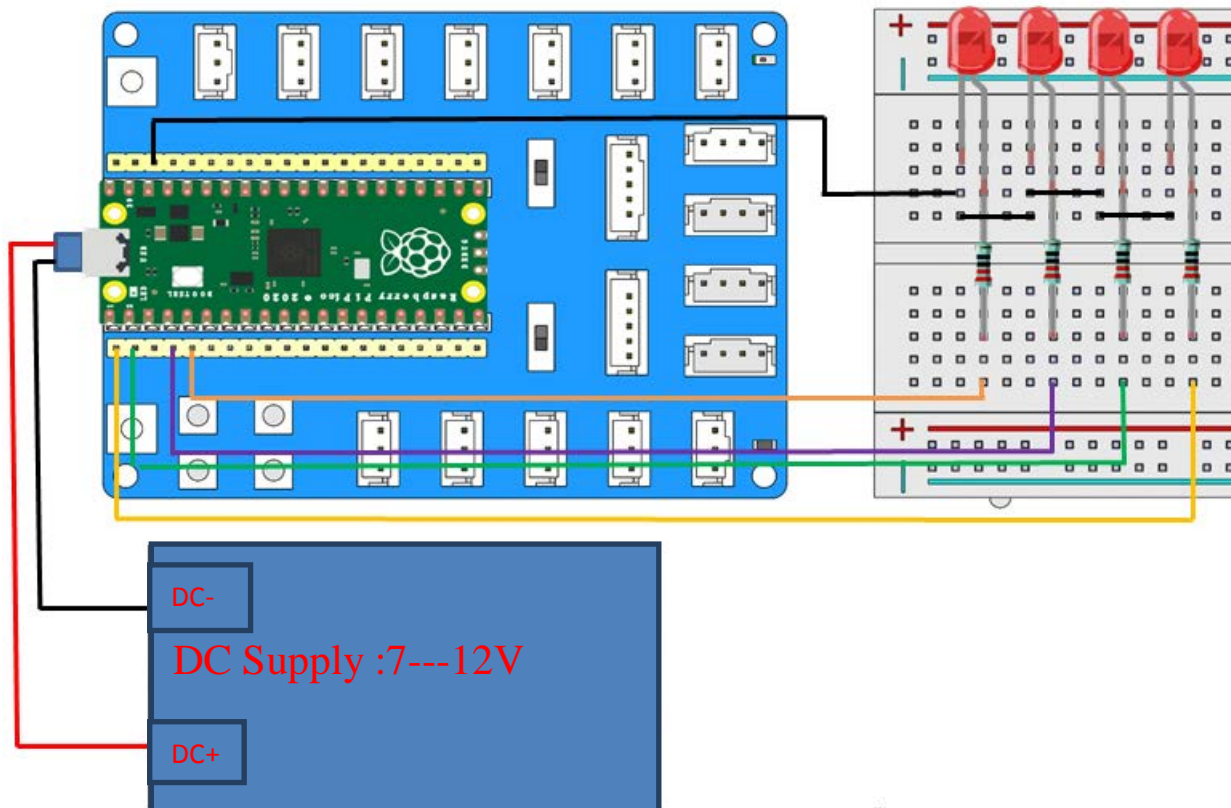
3. Circuit

First, disconnect all power from the Raspberry Pi Pico. Then build the circuit according to the hardware connection diagrams. After the circuit is built and verified correct, connect the PC to Raspberry Pi Pico.

CAUTION: Avoid any possible short circuits (especially connecting 3.3V and GND)!

WARNING: A short circuit can cause high current in your circuit, create excessive component heat and cause permanent damage to your hardware!

Hardware connection:



4. Code

4.1 Arduino code

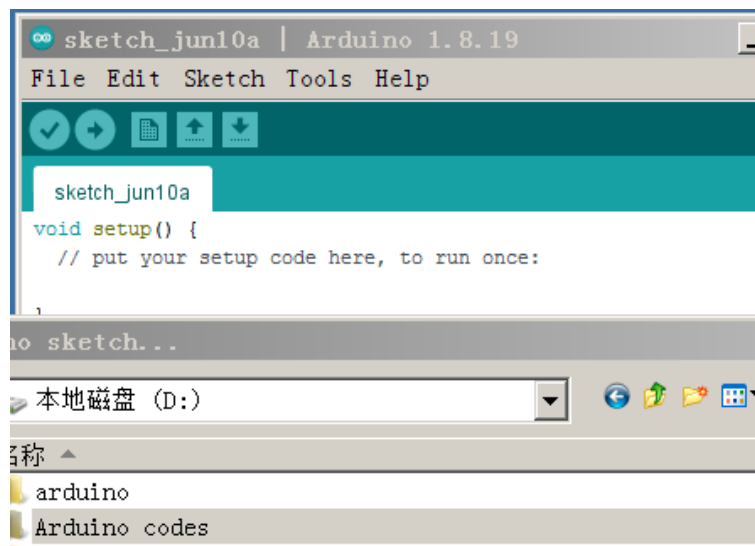
Codes used in this tutorial are saved in "Pico Expansion Kit Tutorial\Lessons for Arduino\Arduino_Codes". You can move the codes to any location. For example, we save the codes in Disk(D) with the path of "D:/ Arduino codes".

Configuring the Operating Environment For Arduino

Please refer to the document "3 Configuring the Operating Environment For Arduino" which saved in "Pico_Extension_Board Kit Tutorial"

Waterfall light led

Open "Arduino" IDE, click "File"--"Open"--"D:"--"Arduino codes".



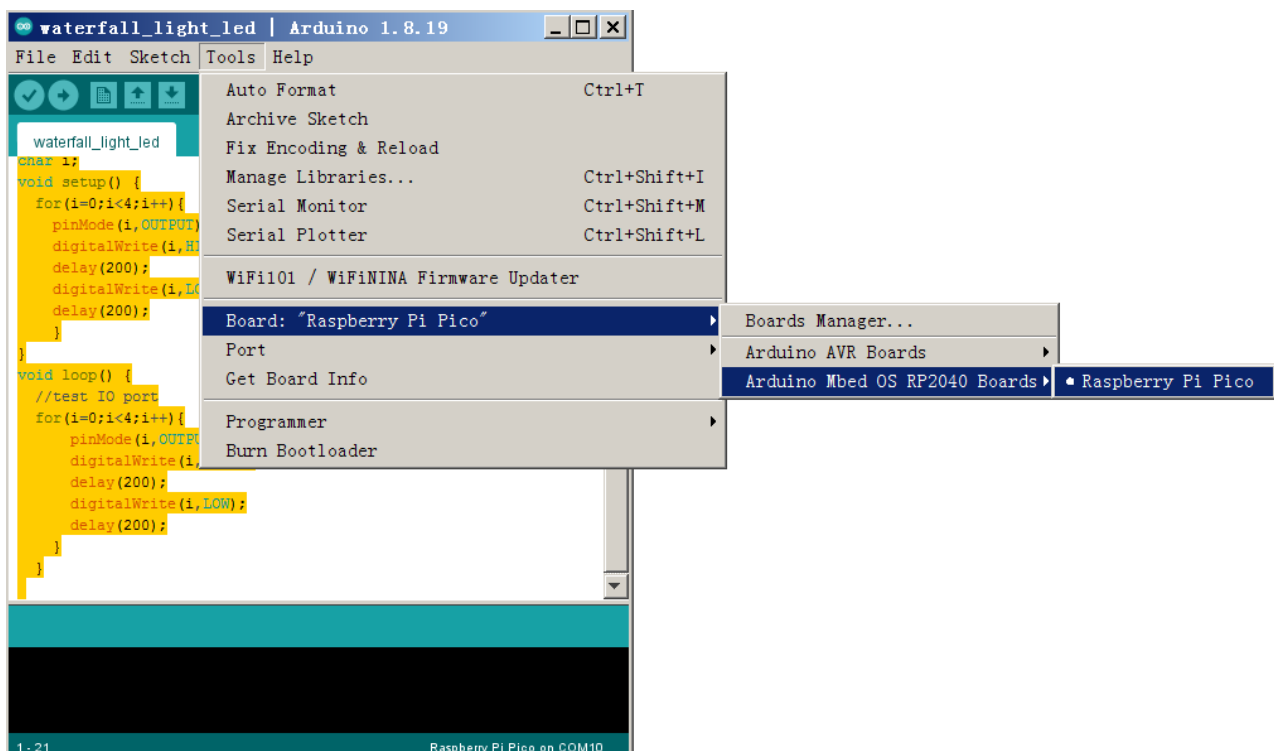
Expand folder “[waterfall_light_led](#)” and double click “[waterfall light led.ino](#)” open it. As shown in the illustration below.

```

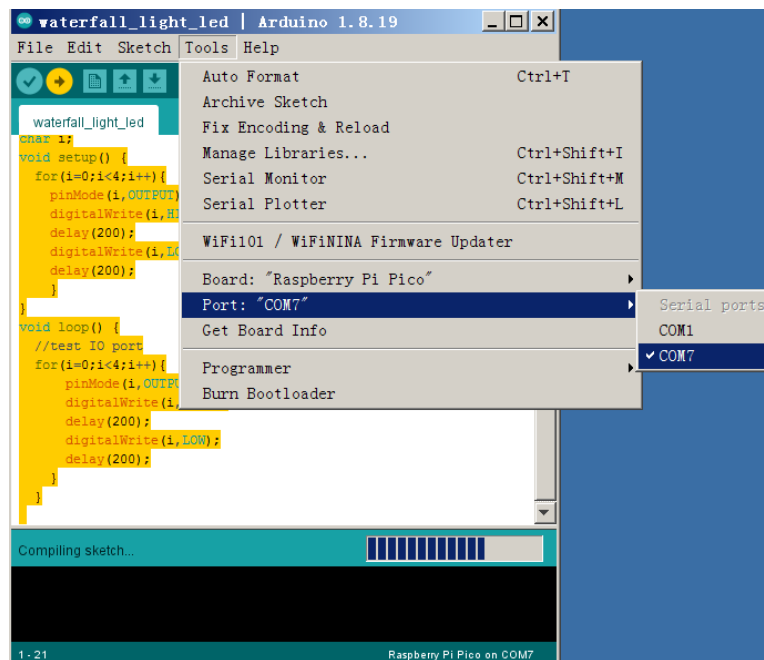
waterfall_light_led | Arduino 1.8.19
File Edit Sketch Tools Help

waterfall_light_led
char i;
void setup() {
  for(i=0;i<4;i++){
    pinMode(i,OUTPUT);
    digitalWrite(i,HIGH);
    delay(200);
    digitalWrite(i,LOW);
    delay(200);
  }
}
void loop() {
  //test IO port
  for(i=0;i<4;i++){
    pinMode(i,OUTPUT);
    digitalWrite(i,HIGH);
    delay(200);
    digitalWrite(i,LOW);
    delay(200);
  }
}
  
```

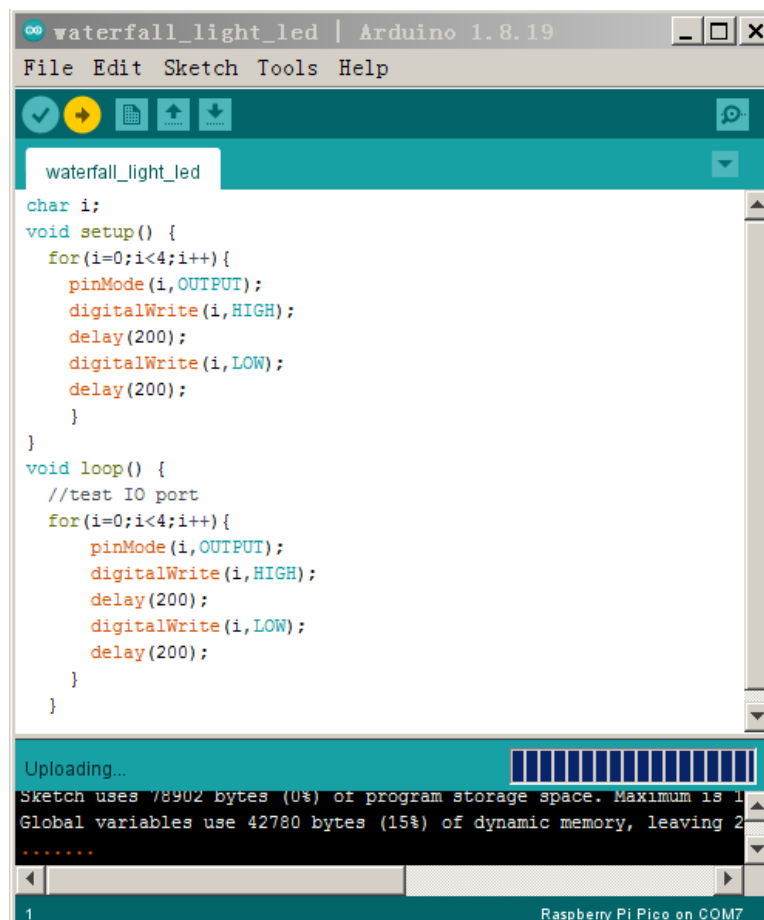
Make sure Raspberry Pi Pico has been connected with the computer. Then click “[Tools](#)”-“[Board](#)”-“[Arduino Mbed OS RP2040 Boards](#)”-“[Raspberry Pi Pico](#)” As shown in the illustration below.



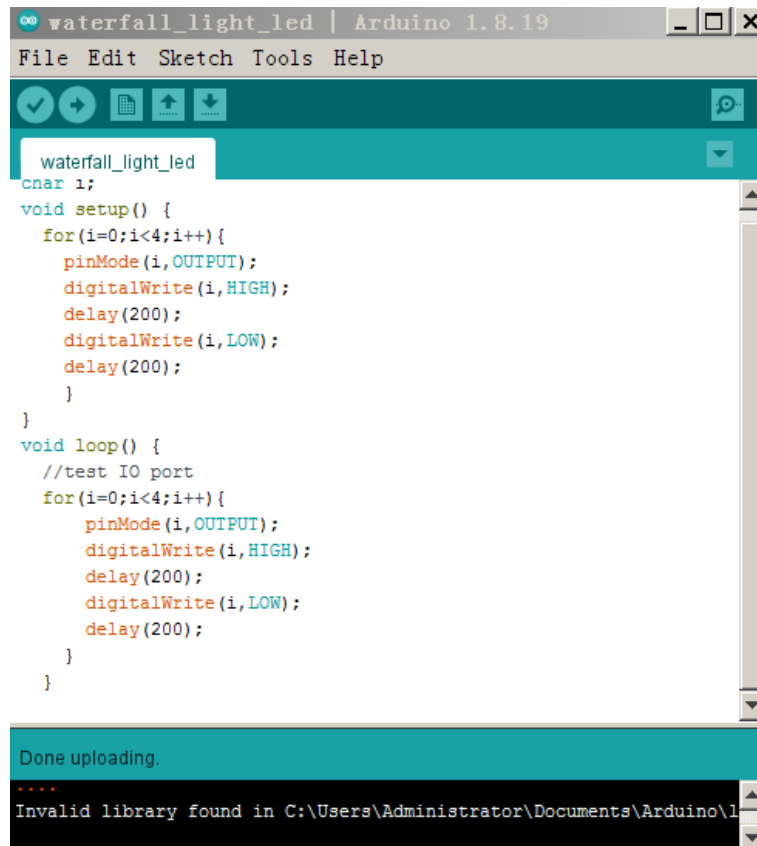
Click “Tools”-“Port” to select the Serial ports for Raspberry Pi Pico .As shown in the illustration below.



Upload the code to Raspberry Pi Pico as shown in the illustration below.



Upload succeeded as shown in the illustration below.



```
waterfall_light_led | Arduino 1.8.19
File Edit Sketch Tools Help

waterfall_light_led
char i;
void setup() {
  for(i=0;i<4;i++){
    pinMode(i,OUTPUT);
    digitalWrite(i,HIGH);
    delay(200);
    digitalWrite(i,LOW);
    delay(200);
  }
}
void loop() {
  //test IO port
  for(i=0;i<4;i++){
    pinMode(i,OUTPUT);
    digitalWrite(i,HIGH);
    delay(200);
    digitalWrite(i,LOW);
    delay(200);
  }
}

Done uploading.
.....
Invalid library found in C:\Users\Administrator\Documents\Arduino\l
```

Note: Be sure to keep pressing the “BOOTSEL” button before powering the Pico, otherwise the firmware will not download successfully at the first time.

Arduino code as below

```
char i;
void setup() {
  for(i=0;i<4;i++){
    pinMode(i,OUTPUT);
    digitalWrite(i,HIGH);
    delay(200);
    digitalWrite(i,LOW);
    delay(200);
  }
}
void loop() {
  //test IO port
  for(i=0;i<4;i++){
    pinMode(i,OUTPUT);
    digitalWrite(i,HIGH);
    delay(200);
    digitalWrite(i,LOW);
    delay(200);
  }
}
```


What's Next?

THANK YOU for participating in this learning experience!

We have reached the end of this Tutorial. If you find errors, omissions or you have suggestions and/or questions about the Tutorial or component contents of this Kit, please feel free to contact us:
cokoino@outlook.com

We will make every effort to make changes and correct errors as soon as feasibly possible and publish a revised version.

If you want to learn more about Arduino, Raspberry Pi, Smart Cars, Robotics and other interesting products in science and technology, please continue to visit our website. We will continue to launch fun, cost-effective, innovative and exciting products.
<http://cokoino.com/>

Thank you again for choosing Cokoino products.